```java

import com.cete.dynamicpdf.*;
import com.cete.dynamicpdf.pageelements.*;
import com.cete.dynamicpdf.pageelements.barcoding.Ean13Sup5;
import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.GregorianCalendar;
import javax.servlet.ServletConfig;
import javax.servlet.ServletException;
import javax.servlet.ServletOutputStream;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class TimeMachine extends HttpServlet {

    ServletOutputStream sOut;
    Template footerTemplate;
    EvenOddTemplate headerTemplate;

    public void init(ServletConfig servletConfig) throws ServletException {
        super.init(servletConfig);

    }
    public void doGet(HttpServletRequest req, HttpServletResponse res)
                     throws IOException,ServletException {


        sOut = res.getOutputStream();
        footerTemplate = new Template();
        headerTemplate = new EvenOddTemplate();
        // Create a document and set it's properties
    Document objDocument = new Document();
    objDocument.setCreator("TimeMachine.java");
    objDocument.setAuthor("H. G. Wells");
    objDocument.setTitle("The Time Machine");
    objDocument.setTemplate(headerTemplate);

    // Adds elements to the header and footer groups
    setPageHeaderTemplate();
    setPageFooterTemplate();

    // Sets up outline hierarchy
    Outline objTitlePageOutline = objDocument.getOutlines().add("The Time "+
                                          "Machine", new XYDestination(1, 0, 0));
```

```
    Outline objChaptersOutline = objTitlePageOutline.getChildOutlines()
                                                        .add("Chapters");

    // Builds the report
    buildDocument(objDocument, objTitlePageOutline, objChaptersOutline);

    // Outputs the document to the current web page
    objDocument.drawToWeb(req, res, sOut, "TimeMachine.pdf");
        sOut.close();
    }

    private void buildDocument(Document document, Outline titlePage,
                               Outline chapters) {
    // Adds Title page to document
    document.getSections().begin(NumberingStyle.NONE, "Cover");
    addTitlePage(document);
    document.getSections().begin(NumberingStyle.NUMERIC, "Page",
                                     footerTemplate);

    // Adds Chapters to the document
    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
                  "_Chapter1.txt"), "1", "Chapter 1", chapters);
    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
        "_Chapter2.txt"), "2", "Chapter 2", chapters);
    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
                  "_Chapter3.txt"), "3", "Chapter 3", chapters);
    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
                  "_Chapter4.txt"), "4", "Chapter 4", chapters);
    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
                  "_Chapter5.txt"), "5", "Chapter 5", chapters);
    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
                  "_Chapter6.txt"), "6", "Chapter 6", chapters);
    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
                  "_Chapter7.txt"), "7", "Chapter 7", chapters);
    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
                  "_Chapter8.txt"), "8", "Chapter 8", chapters);
    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
                  "_Chapter9.txt"), "9", "Chapter 9", chapters);
    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
                  "_Chapter10.txt"), "10", "Chapter 10", chapters);
    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
                  "_Chapter11.txt"), "11", "Chapter 11", chapters);
    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
                  "_Chapter12.txt"), "12", "Chapter 12", chapters);

    // Add the Epilogue to the document
    addChapter(document, getServletContext().getRealPath("/text/TimeMachine"+
                  "_Epilogue.txt"), "Epilogue", "Epilogue", titlePage);
    }
```

```
    private void setPageHeaderTemplate() {

    // Adds elements to the header group
    headerTemplate.getOddElements().add(new Label("The Time Machine",
            -18, -18, 324, 11, Font.getTimesRoman(), 11, TextAlign.LEFT));
    headerTemplate.getOddElements().add(new Label("H. G. Wells", 18, -18,
            324, 11, Font.getTimesRoman(), 11, TextAlign.RIGHT));
        headerTemplate.getEvenElements().add(new Label("H. G. Wells", -18, -18,
            324, 11, Font.getTimesRoman(), 11, TextAlign.LEFT));
    headerTemplate.getEvenElements().add(new Label("The Time Machine", 18,
            -18, 324, 11, Font.getTimesRoman(), 11, TextAlign.RIGHT));
    }

    private void setPageFooterTemplate() {
    // Adds elements to the footer group
    PageNumberingLabel objPageNumberLabel = new PageNumberingLabel
                                                    ("- %%SP%% -", 0, 478, 324,
                                                    11, Font.getTimesRoman(),
                                                    11, TextAlign.CENTER);
    objPageNumberLabel.setPageOffset(-1);
    footerTemplate.getElements().add(objPageNumberLabel);
    }

    private void addTitlePage(Document document) {
    //Adds a title page to the document
    Page objPage = new Page(396, 540, 36);

    String strDisclaimer = "This document is in the public domain. " +
                            "Permission to use, copy, modify, and " +
                            "distribute this document for any purpose and " +
                            "without fee is hereby granted, without any " +
                            "conditions or restrictions. The barcode below " +
                            "is for demonstration purposes only.";

        GregorianCalendar gc = new GregorianCalendar();
        Date date = gc.getTime();
        SimpleDateFormat sd1 = new SimpleDateFormat("yyyy.MM.dd");
        SimpleDateFormat sd2 = new SimpleDateFormat("HH.mm.ss");
    String strGenerated = "Generated by\nDynamicPDF Generator\non "
                           + sd1.format(date) + ",\nat " + sd2.format(date)
                           + " EST";
    objPage.getElements().add(new Label("The Time Machine", 36, 36, 252, 30,
                                Font.getTimesBold(), 30, TextAlign.CENTER));
    objPage.getElements().add(new Label("by H. G. Wells", 36, 96, 252, 22,
                                Font.getTimesBold(), 22, TextAlign.CENTER));
    objPage.getElements().add(new Label("1895", 36, 148, 252, 22,
                                Font.getTimesBold(), 22, TextAlign.CENTER));
        try {
```

```
            objPage.getElements().add(new Image(getServletContext().getRealPath
                                ("/images/DPDFLogo.png"), 62, 208, 0.21f));
        } catch (FileNotFoundException ex) {
            System.err.println("cannot load the image :"+ex);
        }
        objPage.getElements().add(new Label(strGenerated, 132, 208, 182, 54,
                                Font.getTimesRoman(), 11));
    objPage.getElements().add(new Label(strDisclaimer, 36, 276, 252, 65,
                                Font.getTimesRoman(), 11, TextAlign.JUSTIFY));
    objPage.getElements().add(new Ean13Sup5("201234567890", "90000", 82,
                                360));
    objPage.setApplyDocumentTemplate(false);

    document.getPages().add(objPage);
    }

    private void addChapter(Document document, String filePath, String title,
                            String bookmarkText, Outline parentOutline)    {
    // Retrieves the text from the sections file
    String strSectionText = getTextFromFile(filePath);

    // Adds the first page of the section
    Page objPage = addSectionHeaderPage(document, title, bookmarkText,
                                        parentOutline);
    objPage.setApplyDocumentTemplate(false);

    // Creates a TextArea for the sections text
    TextArea objTextArea = new TextArea(strSectionText, 0, 146, 324, 322,
                                        Font.getTimesRoman(), 11);
    objTextArea.setLeading(14);
    objTextArea.setParagraphSpacing(20);
    objPage.getElements().add(objTextArea);
    document.getPages().add(objPage);

    // Creates a TextArea for the overflow text
    objTextArea = objTextArea.getOverflowTextArea(0, 0, 324, 468);

    // Loops until no overflow text is found.
    while (objTextArea != null) {
        // Adds a new page to the document
        objPage = new Page(396, 540, 36);

        objPage.getElements().add(objTextArea);
        // Adds new page to the document
        document.getPages().add(objPage);
        // Creates a TextArea for the overflow text
        objTextArea = objTextArea.getOverflowTextArea();
    }
    }
```

```
    private Page addSectionHeaderPage(Document document, String title,
                                      String bookmarkText,
                                      Outline parentOutline) {
    // Adds the first page of a section to the document
    Page objPage = new Page(396, 540, 36);



    objPage.getElements().add(new Bookmark(bookmarkText, 0, 0,
                                           parentOutline));
    objPage.getElements().add(new Label("The Time Machine", 0, 36, 324, 30,
                                        Font.getTimesBold(), 30,
                                        TextAlign.CENTER));
    objPage.getElements().add(new Label(title, 0, 96, 324, 22,
                                        Font.getTimesBold(), 22,
                                        TextAlign.CENTER));
    objPage.getElements().add(new Line(120, 128, 204, 128));

    return objPage;
    }

    private String getTextFromFile(String filePath) {
        // Opens a text file and returns the text from it.
        StringBuffer contents = new StringBuffer();

        BufferedReader reader = null;
        try {
            reader = new BufferedReader(new FileReader(filePath));
            String line = null;
            while ((line = reader.readLine()) != null){
                contents.append(line);
                contents.append(System.getProperty("line.separator"));
            }
        } catch (FileNotFoundException ex1) {
            System.err.println("Invalid file path :"+ex1);
        } catch (IOException ex2){
            System.err.println("cannot read from the file :"+ex2);
        } finally {
            try {
                reader.close();
            } catch (IOException ex3) {
                System.err.println("cannot close the file :"+ex3);
            }
        }
        return contents.toString();
    }


```

```
}
```