DOCUMENT /RESUME PARSER

GROUP MEMBERS :

Pankaj Shukla -12103049
Ankit Kumar -12103007
Dipesh Kumar -12103019
Rashmi Kandulna -12103062

Summer Training

The Summer Training is an integral part of the B.Tech programme and aims at achieving the following objectives:

- Application of knowledge and techniques learnt to test out and enrich one's understanding, knowledge and skills.
- Gaining deeper understanding in specific functional areas.
- Helps in exploring career opportunities in their areas of interest.

Introduction

- Recruitment agencies usually work with CV Parsing software tools in order to automate the storage and analysis of CV/Resume data. This way, recruiters save hours of manual work by not having to process manually each job application and CV that they receive.
- Resume parser aims to accept a user's resume in Microsoft Word .doc format to reduce the manual step.
- In this phase, we have grouped the titles together to understand the structure of the document.

This is a surprisingly difficult task for a computer to do because:

- Language is infinitely varied. There are hundreds of ways to write down a date, for example, and countless millions of ways to write what you did in your last job.

- A **resume parsing tool** has to capture all these different ways of writing the same thing through complex rules and statistical algorithms.

- Language is ambiguous: the same word or phrase can mean different things in different contexts.

Types of parser

In general, there are three types of approach to parsing a CV/resume:

Keyword based parsers
Grammar based parsers
Statistical parsers

Key measurements

Coverage

Describes WHAT a parser actually tries to extract.

The most advanced CV parsers are even able to extract referees, hobbies, candidate summary, desired salary, desired location, nationality, visa status, and various other fields.

All of this information is required to create a full record for the candidate, so in general the more information a parser extracts, the better.

Accuracy

Describes HOW good a parser is at identifying information from a CV/Resume.

Accuracy measures how often the parser is actually right. This measure is important because the lower the accuracy, the more it costs to correct the errors that the resume parser makes.

In general, if a parser is less than about 90% accurate, the number of errors will be too large to permit it to load data into a resume database without extensive human supervision.

Tools used

- Apache server – Java API for Microsoft Documents
- Eclipse
- PHP
- XML Language

Project Structure

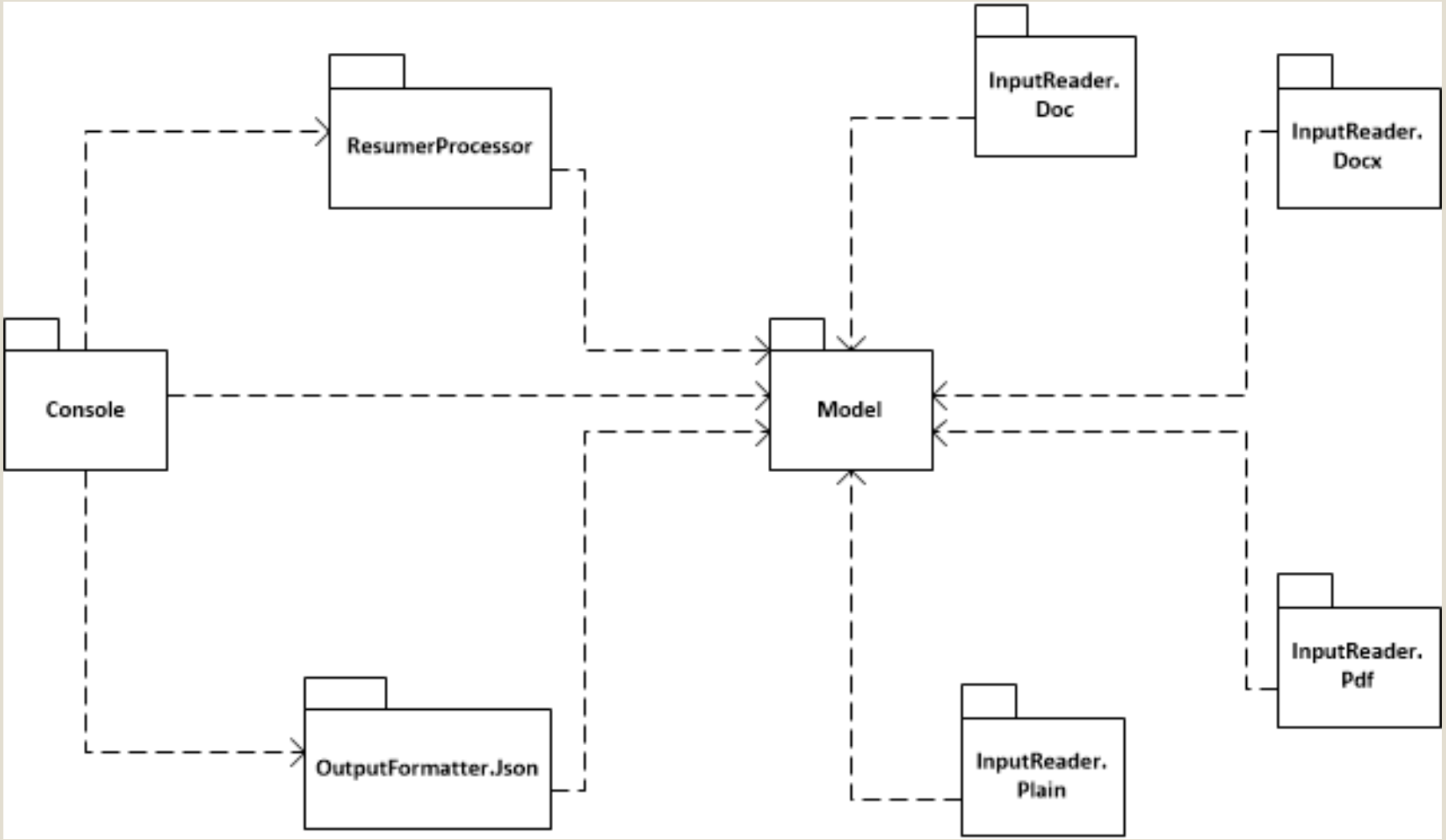
The application solution contains 8 projects:

•**Sharpenter.ResumeParser.UI.Console**: this is the client of the parsing library

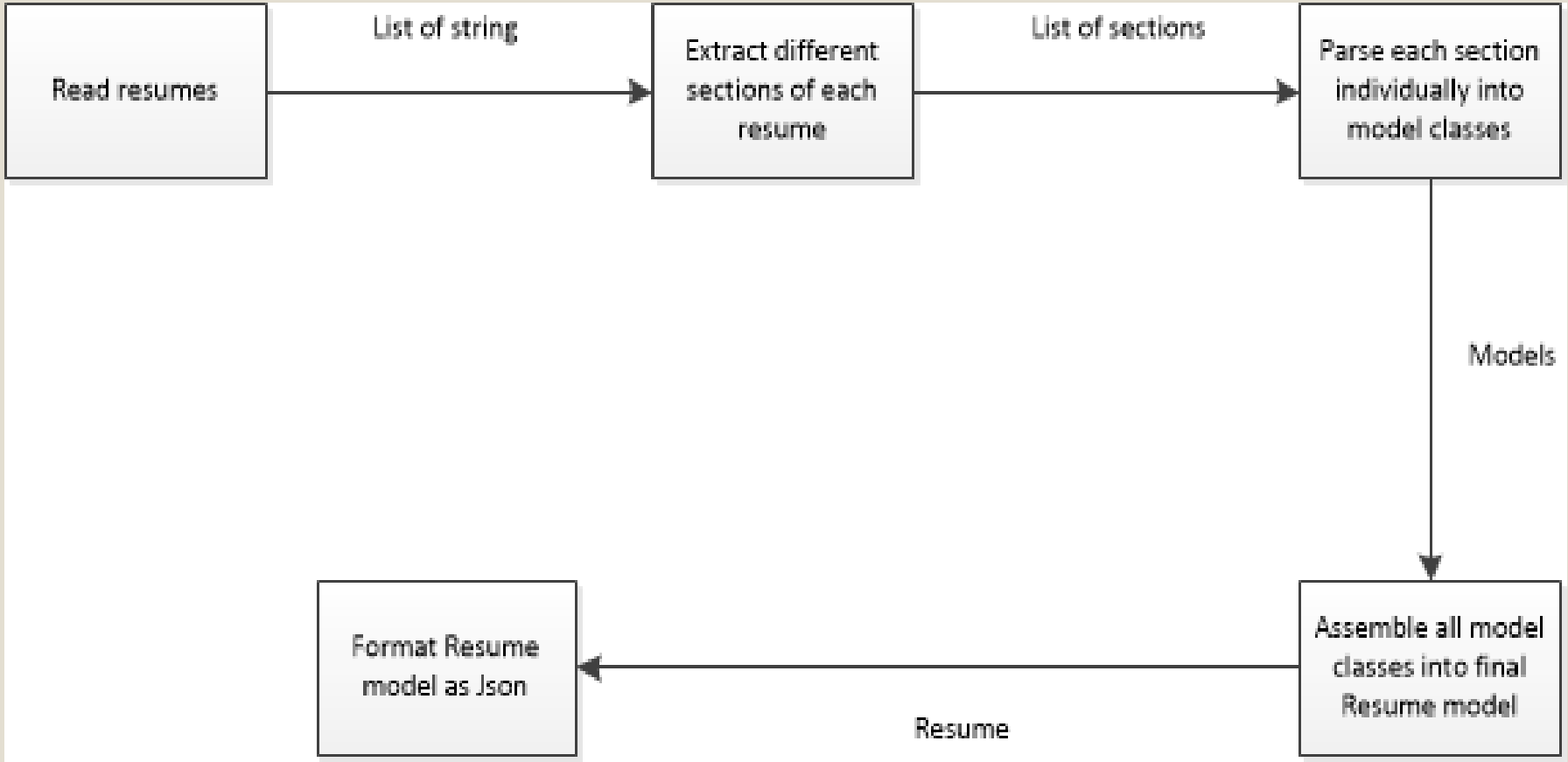
•**Sharpenter.ResumeParser.Model:** this project contains model classes as well as interfaces/contracts used in the whole application

•**Sharpenter.ResumeParser.OutputFormatter.Json:** responsible for serializing model classes to Json for output, convert property to use hyphen convention in Json, etc

•**Sharpenter.ResumeParser.InputReader.Doc:** contain class to read resume in doc file format


ResumerProcessor
InputReader.
Doc
InputReader.
Docx
Console
Model
InputReader.
Pdf
OutputFormatter.Json
InputReader.
Plain

Processing Flow



Processing Flow

- The input resumes will go through several steps before returned as json to client:

- Input readers read resume file from specified location (e.g. folder in hard disk, an url for a html web page) and divide it into list of string

- SectionExtractor will look through the list of string, divide them into different sections (work experience, education, skills, etc)

- Each section is parsed by a parser into object model

- ResumeBuilder assembles all the parsed object models into one complete resume object

- Output formatter takes resume object and formats it before returning to client

Application Design

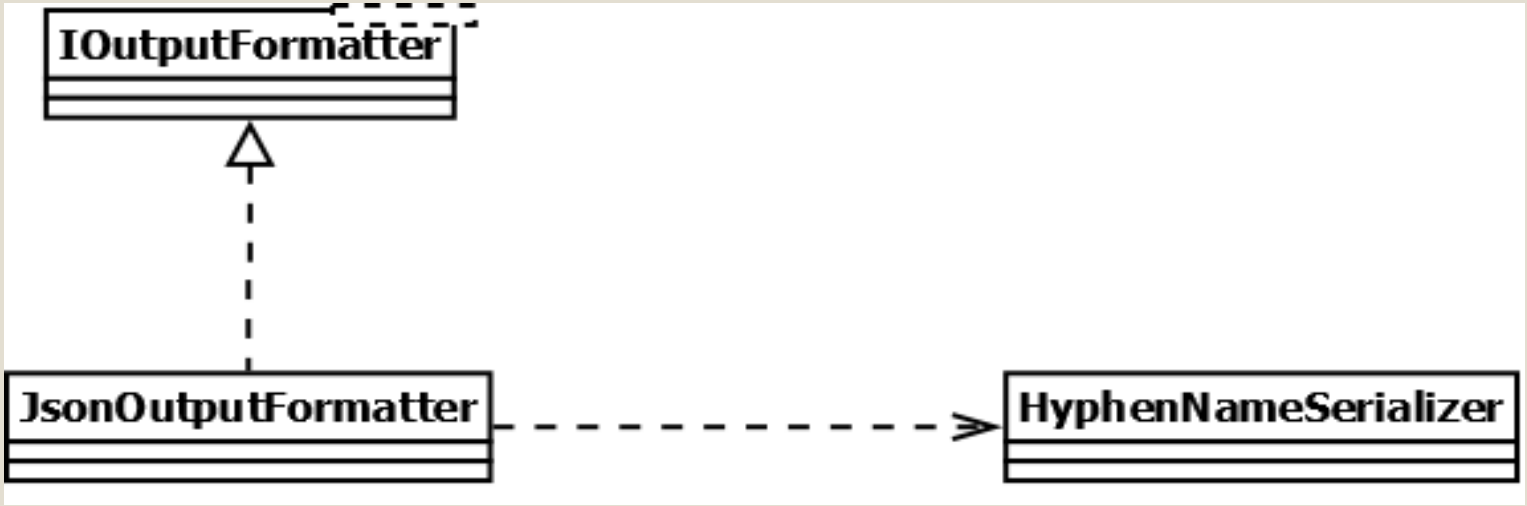
Some highlights of code/application design

•**Output Formatters**

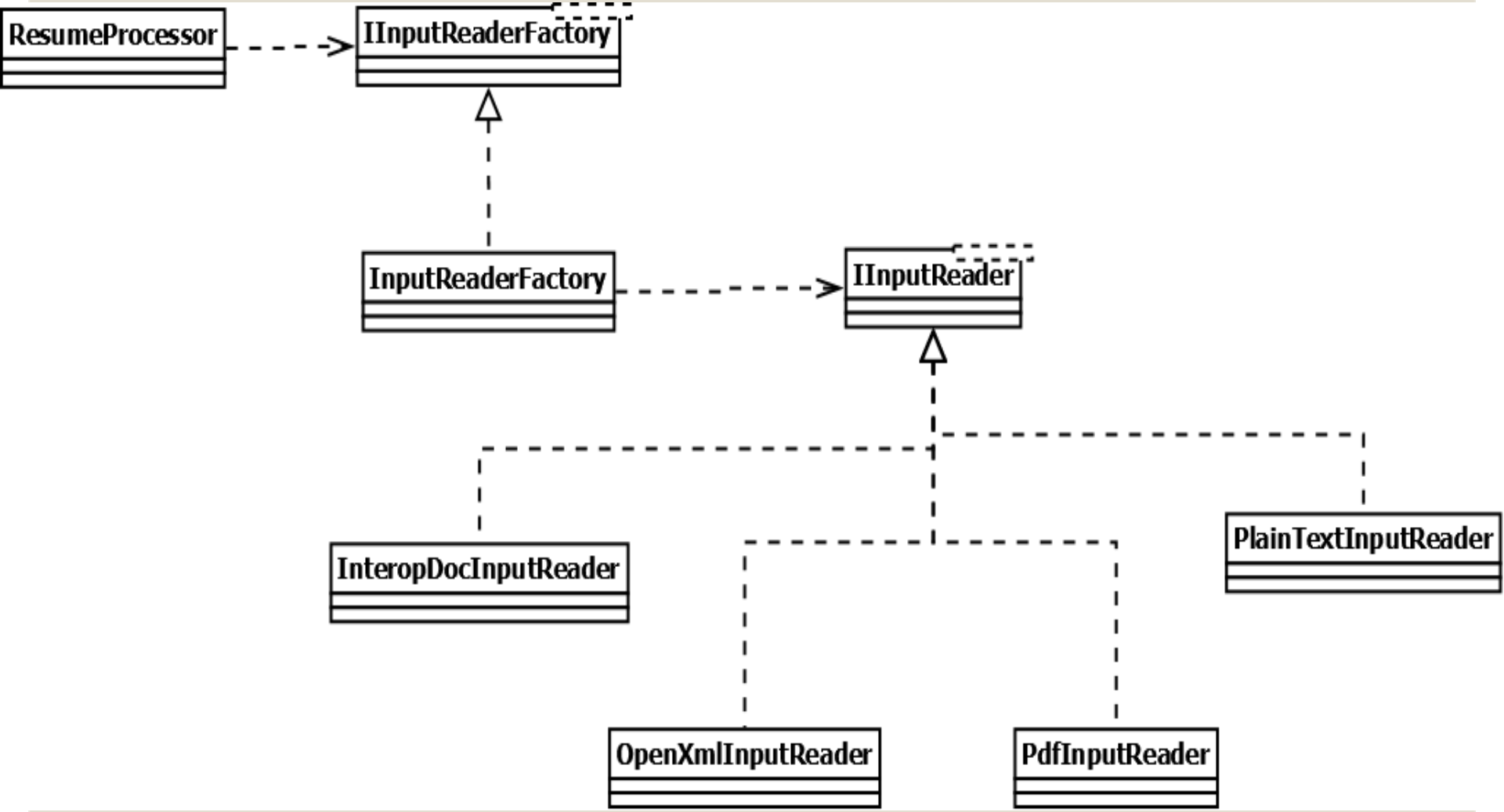
All output formatters implement IOutputFormatter interface from Model project.

```
Public interface Ioutput { string Format (Resume resume); }
```

The application only provides Json output formatter at the moment, but new type of formatter can be added easily.

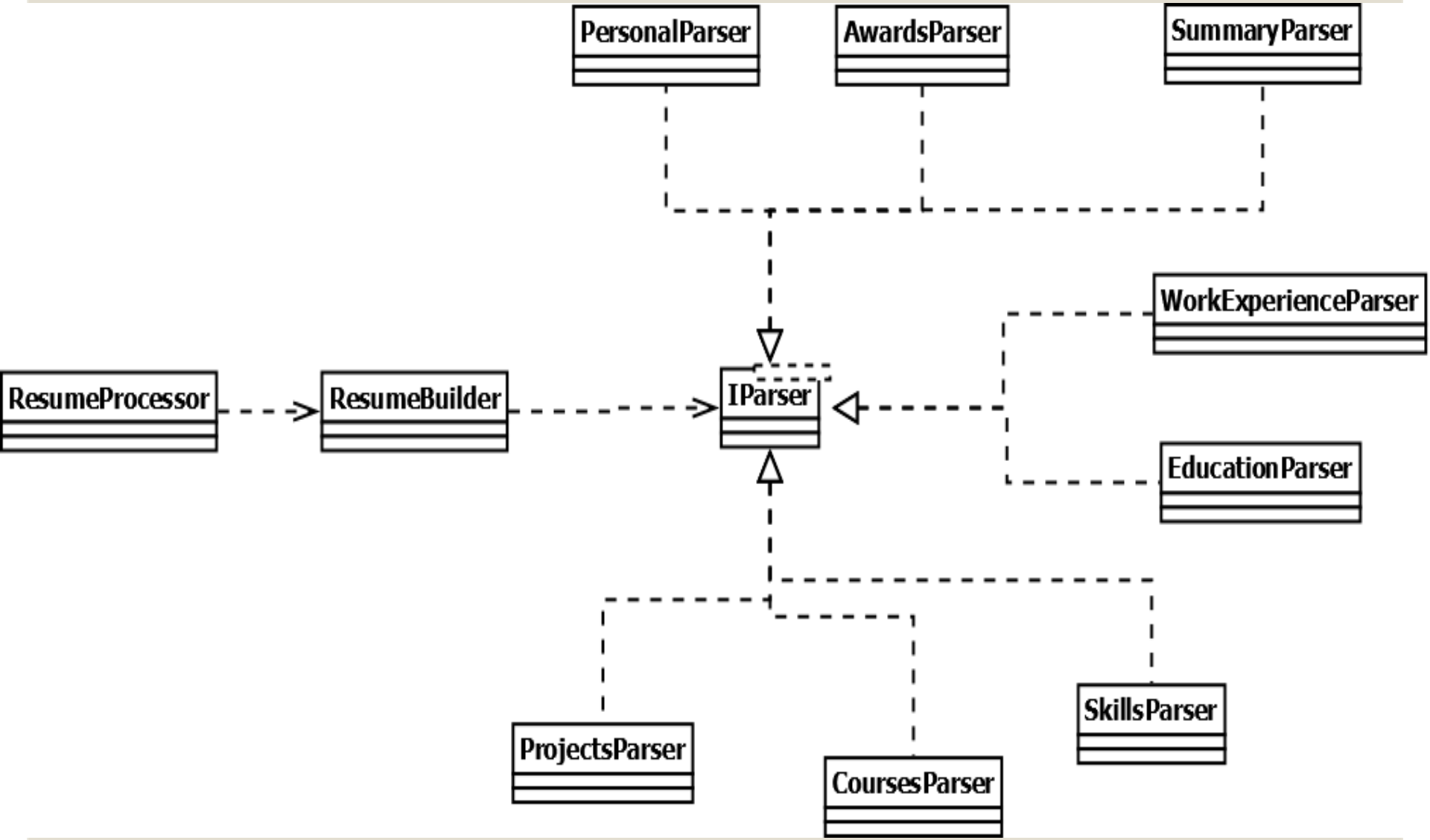


Input Readers



Parsers

After input are divided into different sections, parser classes are called to parse each section into different model class. Each parser class is responsible for one section.


PersonalParser
AwardsParser
SummaryParser
WorkExperienceParser
ResumeProcessor
ResumeBuilder
IParser
EducationParser
SkillsParser
ProjectsParser
CoursesParser

Why we need Resume Parser?

- The process to handle incoming resumes, usually handle by human.
- Depends more on reading capacity of person.
- Time consuming and boring.
- More prone to errors.
- Chances are we miss out resumes
- Or we are too delayed in calling right candidate as his/her resume was either missed or was misread.

Benefits

- Save 85% in your resume reading process
- Huge cost saving per month .
- Have a capacity to read 35 resumes in one minute.
- Capacity to work 24x7
- Very low initial cost to begin using the platform.
- Proper managing the resume data rather than storing them.

Thank You