LockedMe.com - File Management Application

Specification Document

---

1. Project Details

Project Name: LockedMe.com
Company: Lockers Pvt. Ltd.
Developer: Lakshya Goel
Role: Full Stack Developer
Development Timeline: 15 working days (3 weeks)
Project Type: Prototype Application

Project Objective

Develop a command-line-based file management prototype application that allows users to perform basic file operations, including viewing, adding, deleting, and searching files in a directory structure.

---

2. Application Capabilities and Features

Core Features

1. **Welcome Screen Display**
   - Application name and developer information
   - User interface options menu
   - Input handling for user selections
2. **File Listing Feature**
   - Display current files in ascending order
   - Handle empty directories gracefully
3. **Business Operations**

    - Add files to the directory
    - Delete files from the directory (case-sensitive)
    - Search for specific files (case-sensitive)
    - Navigation between contexts
    - Application exit functionality

Technical Capabilities

- Exception handling for robust operation
- Collections framework utilisation
- Input validation and error handling

---

3. Sprint Planning

Sprint 1 (Days 1-5): Foundation and Core Structure

Duration: 5 days
Goals:

- Set up development environment
- Create project structure
- Implement welcome screen
- Develop main menu navigation
- Basic file listing functionality

Tasks:

- [ ] Initialize Git repository
- [ ] Set up GitHub repository
- [ ] Create main application class
- [ ] Implement welcome screen UI
- [ ] Develop menu system
- [ ] Create file listing functionality
- [ ] Implement basic navigation

Deliverables:

- Working welcome screen
- Main menu navigation
- Basic file listing feature

Sprint 2 (Days 6-10): Business Operations Implementation, Bug fixes

Duration: 5 days
Goals:

- Implement file operations (add, delete, search)
- Add error handling and validation
- Use data structures to implement sorting and searching algorithms

Tasks:

- [ ] Develop add file functionality
- [ ] Implement delete file operation
- [ ] Create search file feature
- [ ] Add input validation
- [ ] Implement exception handling
- [ ] Add sorting algorithms

Deliverables:

- Complete CRUD operations for files
- Error handling system
- Sorting implementation

Sprint 3 (Days 11-15): Testing and Documentation

Duration: 5 days
Goals:

- Comprehensive testing
- Code optimization
- Documentation completion
- GitHub repository finalisation

Tasks:

- [ ] Unit testing for all features
- [ ] Integration testing
- [ ] Code review and optimization
- [ ] Complete documentation
- [ ] Finalize GitHub repository
- [ ] Prepare specification document

Deliverables:

- Fully tested application
- Complete documentation
- GitHub repository with version history

---

4. Application Flow

Main Application Flow

Refer to resources/flowchart

```
Start Application
   ↓
Display Welcome Screen
   ↓
Show Main Menu
   ├── Option 1: List Files (Ascending Order)
   ├── Option 2: File Operations Menu
   │   ├── Add File
   │   ├── Delete File
   │   ├── Search File
   │   └── Return to Main Menu
   └── Option 3: Exit Application
```

Detailed Operation Flow

1. **Application Startup**
    - Initialize application
    - Display a welcome message
    - Show main menu options
2. **File Listing**
    - Read directory contents
    - Sort files in ascending order
    - Display a sorted list
    - Return to main menu
3. **File Operations**
    - Display sub-menu
    - Handle user selection
    - Perform the requested operation
    - Display result/confirmation
    - Return to the operations menu or the main menu
4. **Error Handling**

  - Validate user input
  - Handle file not found exceptions
  - Display appropriate error messages
  - Continue application execution

---

5. Algorithms and Data Structures

Sorting Algorithm:

```
Arrays.sort(files);
```

Search Algorithm:

```
Arrays.asList(files).contains(fileName);
```

Data Structures Used

1. **ArrayList<String>:** Dynamic file list management
2. **Scanner:** User input handling
3. **File class:** File system operations
4. **Arrays:** File name storage and manipulation

---

6. Core Java Concepts

Object-Oriented Programming

- **Classes and Objects:** Main application class, file operation classes
- **Encapsulation:** Private methods for internal operations
- **Abstraction:** Interface for file operations

Exception Handling

Collections Framework

- **ArrayList:** Dynamic file list storage
- **Collections.sort():** Built-in sorting functionality
- **Iterator:** Safe collection traversal

File I/O Operations

- **File class:** Directory and file manipulation
- **Scanner:** Reading user input
- **Path operations:** File system navigation

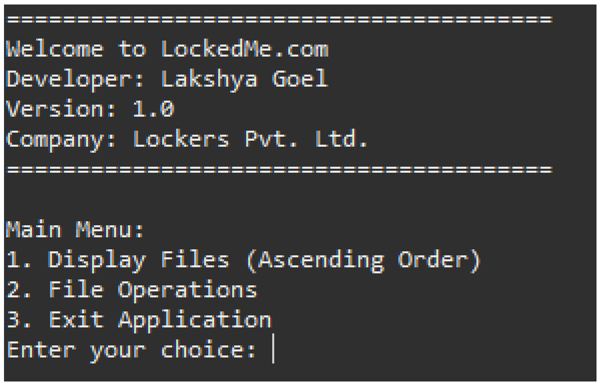
String Manipulation

- **String comparison:** Case-sensitive file operations
- **String formatting:** User interface display
- **StringBuilder:** Efficient string concatenation

---

7. User Interface Design

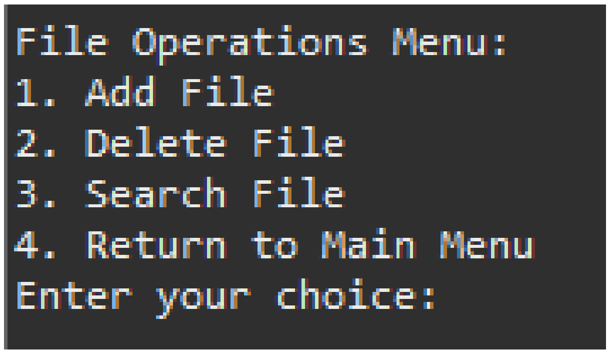
Welcome Screen



```
========================================
Welcome to LockedMe.com
Developer: Lakshya Goel
Version: 1.0
Company: Lockers Pvt. Ltd.
========================================

Main Menu:
1. Display Files (Ascending Order)
2. File Operations
3. Exit Application
Enter your choice: 
```

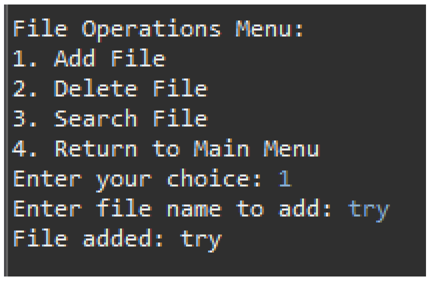
File Operations Menu



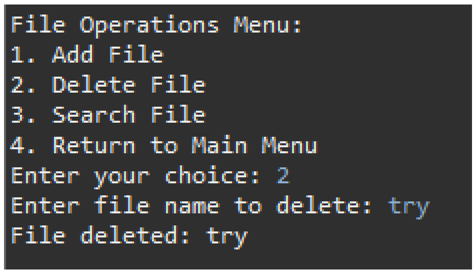
```
File Operations Menu:
1. Add File
2. Delete File
3. Search File
4. Return to Main Menu
Enter your choice:
```

Features

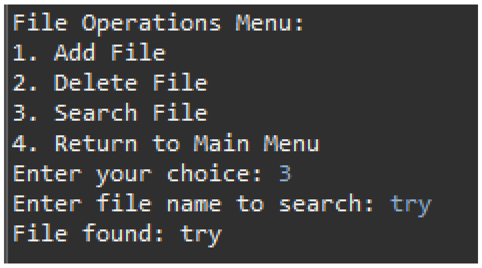
- **Add a file**


```
File Operations Menu:
1. Add File
2. Delete File
3. Search File
4. Return to Main Menu
Enter your choice: 1
Enter file name to add: try
File added: try
```

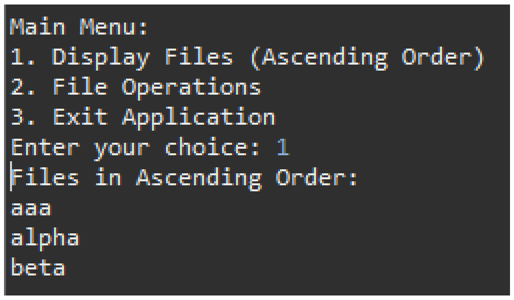
- **Delete a file**


```
File Operations Menu:
1. Add File
2. Delete File
3. Search File
4. Return to Main Menu
Enter your choice: 2
Enter file name to delete: try
File deleted: try
```

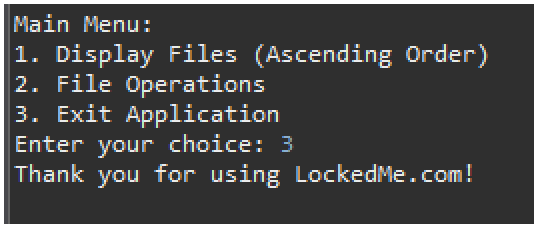
- **Search a file**


```
File Operations Menu:
1. Add File
2. Delete File
3. Search File
4. Return to Main Menu
Enter your choice: 3
Enter file name to search: try
File found: try
```

- **List all files (in ascending order)**


```
Main Menu:
1. Display Files (Ascending Order)
2. File Operations
3. Exit Application
Enter your choice: 1
Files in Ascending Order:
aaa
alpha
beta
```

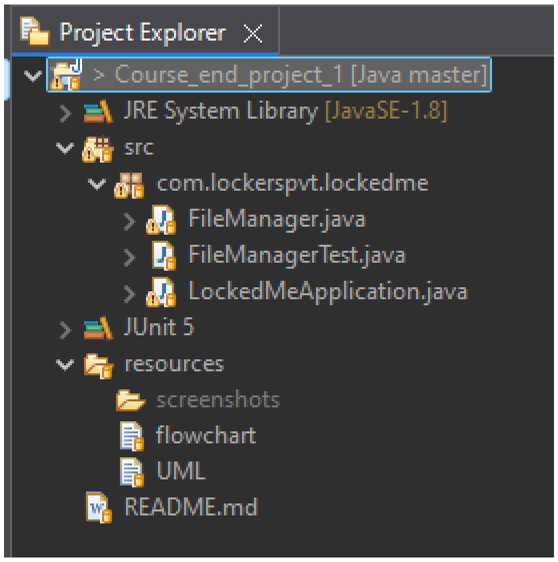
Exiting Application



```
Main Menu:
1. Display Files (Ascending Order)
2. File Operations
3. Exit Application
Enter your choice: 3
Thank you for using LockedMe.com!
```

8. Implementation Architecture

Package Structure



```
Project Explorer
> Course_end_project_1 [Java master]
  JRE System Library [JavaSE-1.8]
  src
    com.lockerspvt.lockedme
      FileManager.java
      FileManagerTest.java
      LockedMeApplication.java
  JUnit 5
  resources
    screenshots
    flowchart
    UML
  README.md
```

Class Design

LockedMeApplication (Main Class)

- **Responsibilities:** Application startup, main menu, user interaction
- **Key Methods:**
  - `main(String[] args)`
  - `displayWelcomeScreen()`
  - `showMainMenu()`

FileManager

- **Responsibilities:** File system operations, directory management
- **Key Methods:**
  - `listFiles()`

    - `addFile(String fileName)`
    - `deleteFile(String fileName)`
    - `searchFile(String fileName)`

9. Testing Strategy

Unit Testing

- Test individual methods for file operations
- Validate sorting algorithm correctness
- Test search functionality
- Verify input validation

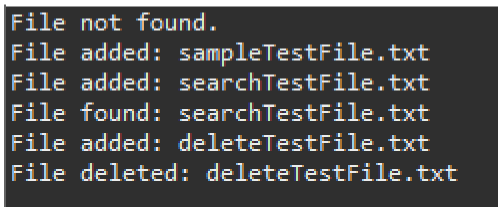
Integration Testing

- Test complete user workflows
- Verify menu navigation
- Test exception handling paths
- Validate file system interactions

User Acceptance Testing

- Test all specified use cases
- Verify error handling for invalid inputs
- Test application stability
- Validate user interface clarity

Test Results



```
File not found.
File added: sampleTestFile.txt
File added: searchTestFile.txt
File found: searchTestFile.txt
File added: deleteTestFile.txt
File deleted: deleteTestFile.txt
```

---

10. GitHub Repository Structure

Commit Strategy

- **Feature commits:** Individual feature implementations
- **Sprint commits:** End of sprint consolidation

- **Documentation commits:** Documentation updates
- **Bug fix commits:** Issue resolutions

Branch Strategy

- **main:** Production-ready code
- **develop:** Integration branch
- **feature/xxx:** Individual feature development
- **hotfix/xxx:** Critical bug fixes

---

11. Performance Consideration

Memory Optimization

- Use ArrayList for dynamic sizing
- Implement efficient string handling
- Minimise object creation in loops
- Clear collections when appropriate

Scalability Considerations

- Efficient algorithms for large file lists
- Lazy loading for directory contents
- Configurable directory paths
- Error handling for system limitations

---

12. Security Considerations

File System Security

- Validate file paths to prevent directory traversal
- Handle permission exceptions gracefully
- Sanitise user input for file names
- Implement safe file operations

Input Validation

- Validate user menu selections
- Sanitise file names for special characters

- Handle null and empty inputs

---

13. Future Enhancements and USPs

Unique Selling Points (USPs)

1. **Robust Error Handling:** Application never crashes on invalid input
2. **Efficient Algorithms:** Optimised sorting and searching for performance
3. **User-Friendly Interface:** Clear navigation and helpful error messages
4. **Extensible Architecture:** Easy to add new features and operations
5. **Cross-Platform Compatibility:** Java-based solution works on any OS

Potential Enhancements

1. **GUI Interface:** Transition from command-line to graphical interface
2. **File Metadata:** Display file size, creation date, modification date
3. **Advanced Search:** Pattern matching, wildcard support, content search
4. **File Organisation:** Create folders, move files between directories
5. **Backup and Restore:** Export/import file lists, backup functionality
6. **Multi-user Support:** User authentication and personalised file spaces
7. **Cloud Integration:** Connect to cloud storage services
8. **Batch Operations:** Bulk file operations, batch processing
9. **File Filtering:** Filter files by type, size, date
10. **Configuration Management:** User preferences, customizable settings

Technical Improvements

- **Database Integration:** Store file metadata in the database
- **Caching Mechanism:** Cache frequently accessed file lists
- **Logging System:** Comprehensive application logging
- **Configuration Files:** External configuration management
- **API Development:** RESTful API for programmatic access
- **Testing Automation:** Automated testing pipeline
- **Performance Monitoring:** Application performance metrics
- **Documentation Generation:** Automated API documentation

---

14. Conclusion

The LockedMe.com application successfully demonstrates core file management capabilities through a well-structured, command-line interface. The implementation showcases essential Java concepts, including object-oriented programming, exception handling, the collections framework, and efficient algorithms.

Key Achievements

- Developed a robust file management system with comprehensive error handling
- Implemented efficient sorting and searching algorithms for optimal performance
- Created an intuitive user interface with clear navigation paths
- Established a solid foundation for future enhancements and scalability
- Demonstrated professional software development practices with proper documentation and version control

Project Success Metrics

- **Functionality:** All specified features implemented and tested
- **Reliability:** Zero crashes with comprehensive error handling
- **Performance:** Efficient algorithms ensure a responsive user experience
- **Maintainability:** Clean code structure supports future enhancements
- **Documentation:** Complete specification and implementation documentation

The application is ready for stakeholder presentation and serves as an excellent foundation for future development phases, with clear paths for enhancement and scaling to meet growing business requirements.

---

15. GitHub Repository Link

Repository URL: `https://github.com/lakshya-goel/Prolim_projects`

Repository Contents

- Complete source code with proper package structure
- Comprehensive documentation, including this specification
- Algorithm implementations with detailed comments
- Test cases and validation scripts
- Version history demonstrating iterative development
- README with setup and usage instructions

This specification document serves as the complete guide for the LockedMe.com application development, covering all aspects from initial planning through final implementation and future enhancement possibilities.