ClubConnect - Quick Reference Card

Files Provided ✅

Core Files (Ready to Use)

1. **ClubConnectApp.java** - Main entry point
2. **DatabaseManager.java** - Database setup & connection
3. **LoginFrame.java** - Login UI
4. **RegistrationFrame.java** - User registration UI

Models (Entities)

5. **User.java** - User entity with roles
6. **Club.java** - Club entity
7. **Event.java** - Event entity

Data Access Layer

8. **UserDAO.java** - User database operations
9. **ClubDAO.java** - Club database operations
10. **EventDAO.java** - Event database operations

Utilities

11. **SessionManager.java** - Session tracking
12. **PasswordHasher.java** - Password security
13. **Validator.java** - Input validation

UI Examples

14. **MemberDashboard.java** - Complete dashboard example

Documentation

15. **PROJECT_STRUCTURE.md** - Complete project layout
16. **IMPLEMENTATION_GUIDE.md** - Step-by-step instructions
17. **README.md** - Setup & user guide
18. **QUICK_REFERENCE_CARD.md** - This file

What You Need to Create in Cursor

Priority 1: Core Models (30 min)

☐ ClubMember.java
☐ RSVP.java
☐ Resource.java
☐ BudgetTransaction.java
☐ Announcement.java
☐ Discussion.java
☐ Notification.java

Priority 2: Remaining DAOs (2 hours)

☐ ClubMemberDAO.java
☐ RSVPDAO.java
☐ ResourceDAO.java
☐ BudgetTransactionDAO.java
☐ AnnouncementDAO.java
☐ DiscussionDAO.java
☐ NotificationDAO.java

Priority 3: Service Layer (3 hours)

☐ NotificationService.java (with multi-threading)
☐ ClubService.java
☐ EventService.java
☐ MembershipService.java
☐ BudgetService.java
☐ ReportService.java

Priority 4: Dashboards (4 hours)

☐ AdminDashboard.java
☐ LeaderDashboard.java
☐ GuestDashboard.java
(MemberDashboard.java already provided)

Priority 5: UI Panels (3 hours)

☐ ClubListPanel.java
☐ EventCalendarPanel.java

```
☐ MembershipPanel.java
☐ BudgetPanel.java
☐ AnnouncementPanel.java
☐ ReportPanel.java
```

Priority 6: Dialogs (2 hours)

```
☐ CreateClubDialog.java
☐ CreateEventDialog.java
☐ AttendanceDialog.java
☐ BudgetRequestDialog.java
☐ MemberSearchDialog.java
```

Priority 7: Utilities (2 hours)

```
☐ EmailSender.java
☐ CSVExporter.java
☐ PDFGenerator.java
```

Cursor Prompt Templates

For Models:

```
Create [ModelName].java in package com.clubconnect.models with fields:
- [field1]: [type]
- [field2]: [type]
Include constructor, getters, setters, toString. Follow User.java pattern.
```

For DAOs:

```
Create [Entity]DAO.java in com.clubconnect.dao following UserDAO.java pattern with:
- create[Entity]([Entity] obj): boolean
- get[Entity]ById(int id): [Entity]
- getAll[Entities](): List<[Entity]>
- update[Entity]([Entity] obj): boolean
- delete[Entity](int id): boolean
Use PreparedStatements, include exception handling.
```

For Services:

```
Create [Feature]Service.java in com.clubconnect.services with methods:
- [method1 description]
- [method2 description]
Use [Entity]DAO, include validation, error handling, and multi-threading where needed.
```

For UI:

```
Create [Component].java in com.clubconnect.ui.[package] extending J[Frame/Panel/Dialog] with:
- Layout: [GridBagLayout/BorderLayout]
- Components: [JTable/JButton/JTextField/etc.]
- Actions: [onClick handlers]
Include proper event handlers and JOptionPane feedback.
```

Key Packages

```
com.clubconnect
├── ClubConnectApp.java
├── database/
│   └── DatabaseManager.java
├── models/
│   └── [7 model classes]
├── dao/
│   └── [10 DAO classes]
├── services/
│   └── [6 service classes]
├── ui/
│   ├── [Login, Registration, 4 Dashboards]
│   ├── panels/
│   │   └── [6 panel classes]
│   └── dialogs/
│       └── [5 dialog classes]
└── utils/
    └── [6 utility classes]
```

Multi-Threading Locations

Must implement threads in:

1. **NotificationService** - Batch sending
2. **DatabaseManager** - Export on close
3. **EventService** - Reminders background task
4. **DiscussionBoard** - Polling for updates

Example:

```java
Thread task = new Thread(() -> {
    // Your code here
});
task.start();
```

Database Quick Commands

```sql
-- Check database
SHOW DATABASES;
USE clubconnect_db;
SHOW TABLES;

-- Create admin user
INSERT INTO users (student_id, username, email, password, role)
VALUES ('ADMIN001', 'Admin', 'admin@uni.edu', 'HASHED_PASSWORD', 'ADMIN');

-- Check data
SELECT * FROM users;
SELECT * FROM clubs;
SELECT * FROM events;
```

Common Issues & Fixes

Issue	**Solution**
Can't connect to DB	Check MySQL running, verify credentials
ClassNotFoundException	Add JAR files to classpath
UI not showing	Call setVisible(true), check EDT thread
NullPointerException	Add null checks, initialize objects
SQL Exception	Check table exists, verify column names

Testing Checklist

```
☐ Login with admin/member/leader
☐ Register new user
☐ Create club (pending approval)
```

☐ Approve club (admin)
☐ Create event with resource
☐ Check resource conflicts
☐ Request budget for event
☐ Approve budget (admin)
☐ Apply to join club
☐ Approve membership (leader)
☐ RSVP to event
☐ Mark attendance
☐ Send announcement
☐ Generate report
☐ Export to CSV/PDF
☐ Test multi-threading (notifications)
☐ Database export on close

Mark Allocation (Section A: 70 marks)

Component	**Marks**	**Status**
Authentication & Roles	10	✅ Provided
Club Management	10	✅ DAO provided
Membership Handling	8	Need to create
Event Planning	9	✅ DAO provided
Budget Tracking	7	Need to create
Communication	6	Need to create
Reporting	7	Need to create
Data Persistence	5	✅ Provided
Error Handling	3	Partially done
Look & Feel	5	✅ Examples provided

Time Estimates

- Models: 30 minutes
- DAOs: 2 hours
- Services: 3 hours
- Dashboards: 4 hours
- Panels: 3 hours
- Dialogs: 2 hours

- Utilities: 2 hours
- Testing: 3 hours

Total: ~20 hours of focused work

Essential Libraries to Download

1. **mysql-connector-java-8.0.33.jar**
   - https://dev.mysql.com/downloads/connector/j/
2. **itextpdf-5.5.13.3.jar**
   - https://mvnrepository.com/artifact/com.itextpdf/itextpdf/5.5.13.3
3. **javax.mail.jar** (1.4.7)
   - https://mvnrepository.com/artifact/javax.mail/mail/1.4.7

Git Ignore (if using version control)

```
*.class
*.jar (except in lib/)
database_backup.sql
.idea/
out/
target/
*.iml
```

Final Submission Checklist

```
☐ All 70 marks requirements implemented
☐ Multi-threading in 4+ places
☐ Database auto-creates and exports
☐ All roles (Admin/Leader/Member/Guest) work
☐ Events with resource booking functional
☐ Budget request workflow complete
☐ Reports generate and export
☐ Error handling throughout
☐ Professional UI with consistent styling
☐ Project compiles without errors
☐ Executable JAR created
☐ README with setup instructions
☐ Database exported and included
```

☐ All external JARs in project folder

☐ Similarity index < 20%

Emergency Shortcuts

If short on time, focus on these high-mark items:

1. Complete all DAOs (enables database operations)
2. Finish AdminDashboard (shows full functionality)
3. LeaderDashboard with event creation
4. NotificationService with threading
5. ReportService with PDF export
6. Resource booking conflict checking

Resources

- Java Swing Tutorial: https://docs.oracle.com/javase/tutorial/uiswing/
- MySQL JDBC Guide: https://dev.mysql.com/doc/connector-j/en/
- iText Documentation: https://itextpdf.com/resources
- Thread examples in IMPLEMENTATION_GUIDE.md
- UI patterns in MemberDashboard.java

Need Help?

1. Check IMPLEMENTATION_GUIDE.md for detailed examples
2. Review provided code files for patterns
3. Use Cursor with specific prompts from this card
4. Test incrementally (don't wait until end)
5. Ask specific questions about implementations

Remember: You have ~20 hours of work ahead. Stay organized, follow the priority order, and test each component before moving forward. Good luck! 🚀